



Computer Networks

CS3611

Application Layer-Part 2

Haiming Jin

The slides are adapted from those provided by Prof. Romit Roy Choudhury.

Chapter 2: Application layer

- ❑ 2.1 Principles of network applications
- ❑ 2.2 Web and HTTP
- ❑ 2.3 FTP
- ❑ 2.4 Electronic Mail
 - ❖ SMTP, POP3, IMAP
- ❑ 2.5 DNS

DNS: Domain Name System

People: many identifiers:

- ❖ 身份证号, name, passport #

Internet hosts, routers:

- ❖ IP address (32 bit) - used for addressing datagrams
- ❖ “name”, e.g., www.baidu.com - used by humans

Q: map between IP addresses and name ?

Domain Name System:

- ❑ *distributed database* implemented in hierarchy of many *name servers*
- ❑ *application-layer protocol* host, name servers to communicate to *resolve* names (address/name translation)
 - ❖ note: core Internet function, implemented as application-layer protocol
 - ❖ complexity at network’s “edge”

DNS: Domain Name System

- ❑ Imagine a world without DNS

- ❑ You would have to remember the IP addresses of
 - ❖ Every website you want to visit
 - ❖ Your bookmarks will be a list of IP addresses

- ❖ You will speak like

*“I went to 167.33.24.10, and there was an awesome
link to 153.11.35.81... “*

DNS

DNS services

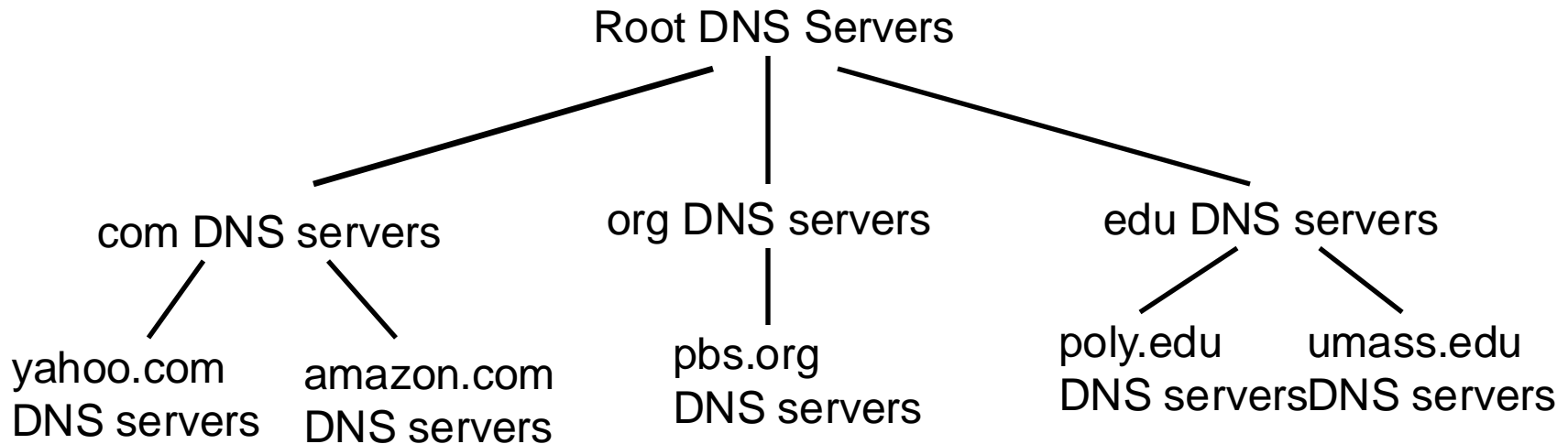
- ❑ Hostname to IP address translation
- ❑ Host aliasing
 - ❖ Canonical and alias names
- ❑ Mail server aliasing
- ❑ Load distribution
 - ❖ Replicated Web servers: set of IP addresses for one canonical name

Why not centralize DNS?

- ❑ single point of failure
- ❑ traffic volume
- ❑ distant centralized database

doesn't *scale!*

Distributed, Hierarchical Database



Client wants IP for www.baidu.com; 1st approx:

- ❑ Client queries a root server to find [.com](http://www.com) DNS server
- ❑ Client queries [com](http://www.com) DNS server to get [baidu.com](http://www.baidu.com) DNS server
- ❑ Client queries [baidu.com](http://www.baidu.com) DNS server to get IP address for www.baidu.com

DNS: Root name servers

- official, contact-of-last-resort by name servers that can not resolve name
- ICANN (Internet Corporation for Assigned Names and Numbers) manages root DNS domain

13 logical root name
“servers” worldwide
each “server”
replicated many
times (~400 around
the world)

TLD and Authoritative Servers

□ Top-level domain (TLD) servers:

- ❖ responsible for com, org, net, etc.

□ Authoritative DNS servers:

- ❖ An organization's DNS servers,
 - providing authoritative hostname to IP mappings for organization's servers (e.g., Web and mail).
- ❖ Can be maintained by organization or service provider

Local Name Server

- Does not strictly belong to hierarchy

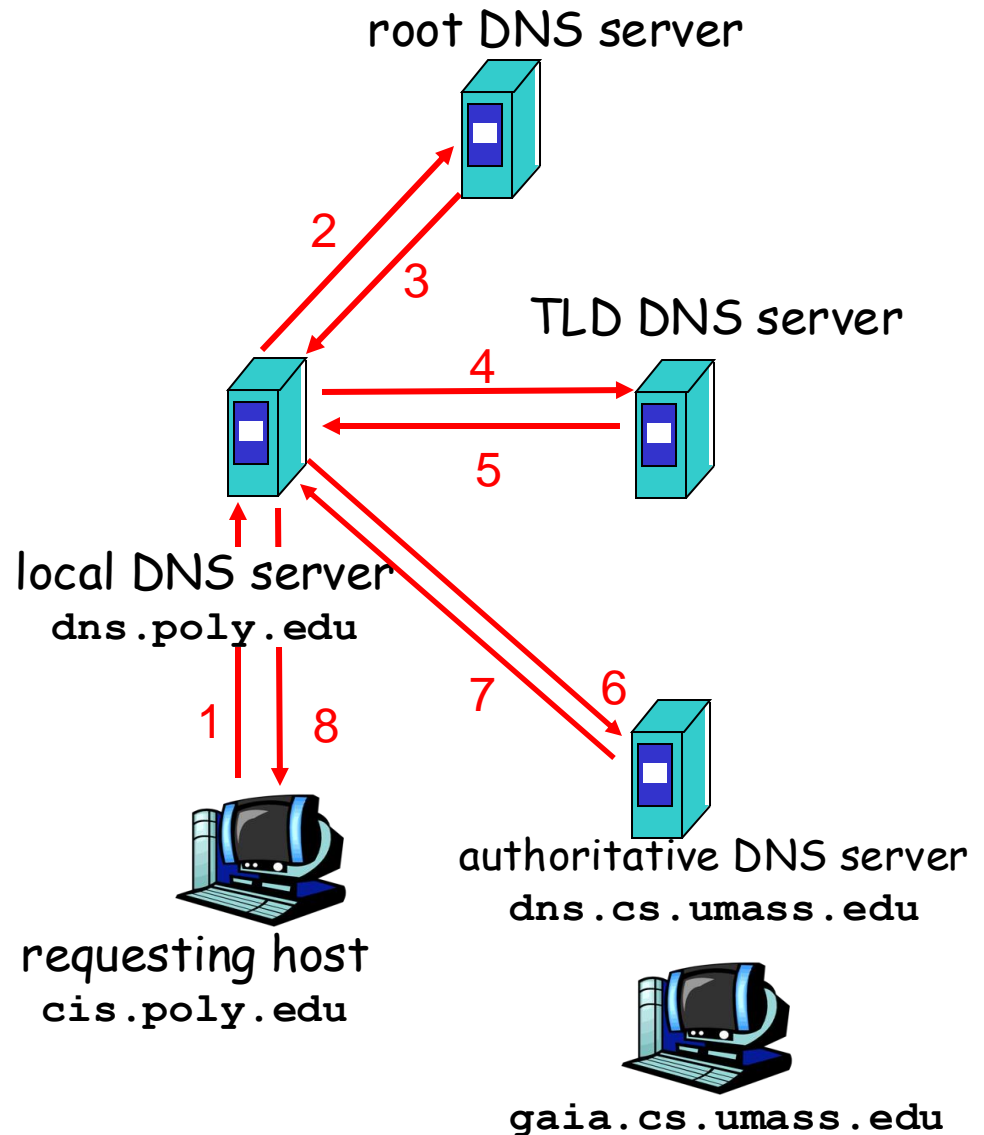
- Each ISP (residential, company, univ) has one.
 - ❖ Also called “default name server”

- When a host makes a DNS query
 - ❖ query is sent to its local DNS server
 - ❖ Acts as a proxy, forwards query into hierarchy.

Example

□ Iterative Querying

Host at cis.poly.edu
wants IP address for
gaia.cs.umass.edu



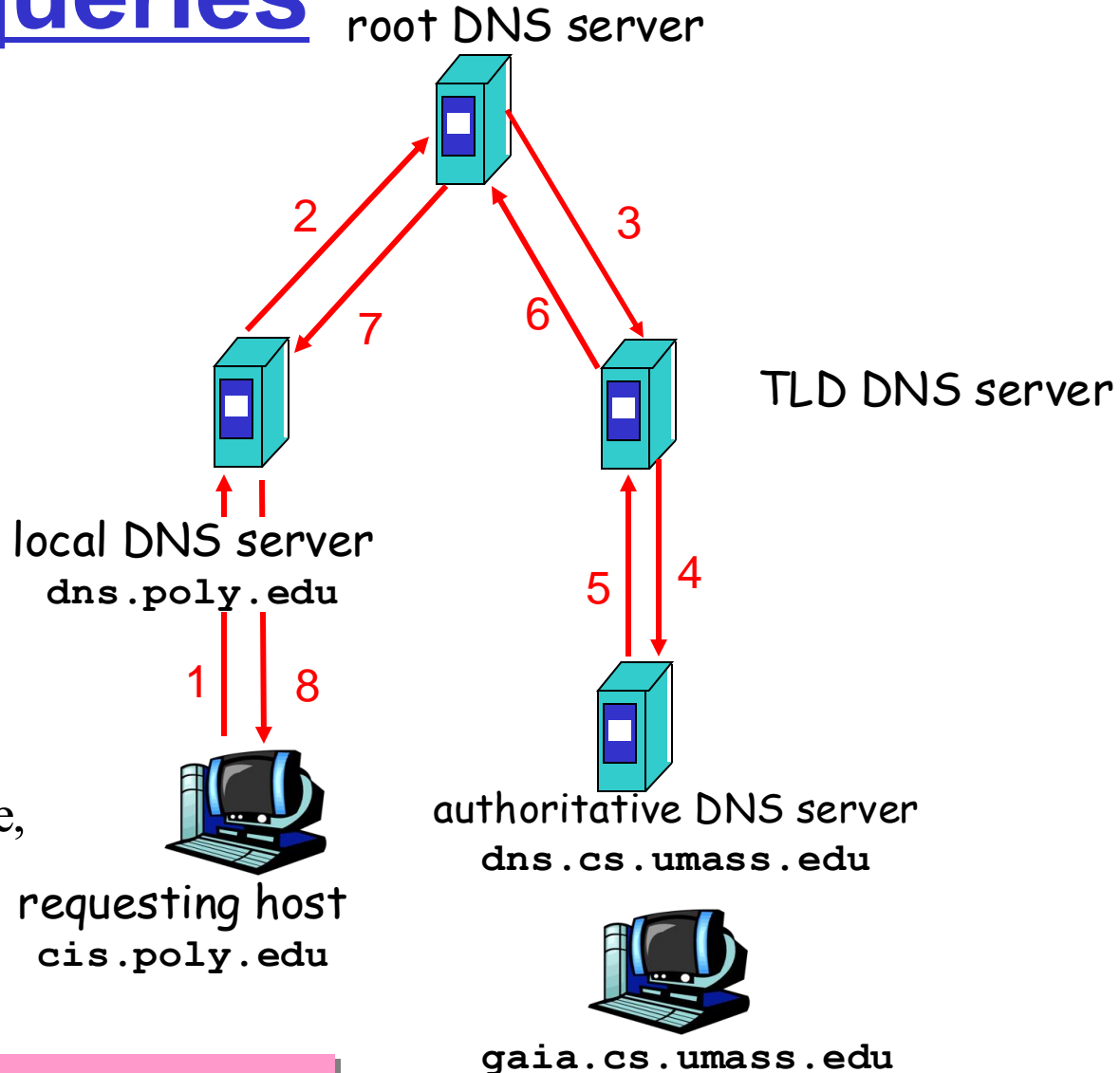
Recursive queries

recursive query:

- ❑ puts burden of name resolution on contacted name server
- ❑ heavy load?

iterative query:

- ❑ contacted server replies with name of server to contact
- ❑ “I don’t know this name, but ask this server”



Which is a better design choice?

DNS: caching

- Once (any) name server learns mapping, it *caches* mapping
 - ❖ cache entries timeout (disappear) after some time
 - ❖ TLD servers typically cached in local name servers
 - Thus root name servers not often visited

DNS records

DNS: distributed db storing resource records (**RR**)

RR format: (name, value, type, ttl)

□ Type=A

- ❖ **name** is hostname
- ❖ **value** is IP address

□ Type=NS

- ❖ **name** is domain (e.g. foo.com)
- ❖ **value** is hostname of authoritative name server for this domain

□ Type=CNAME

- ❖ **name** is alias name for some “canonical” (the real) name
www.ibm.com is really
servereast.backup2.ibm.com
- ❖ **value** is canonical name

□ Type=MX

- ❖ **value** is name of mailserver associated with **name**

DNS protocol, messages

DNS protocol : *query* and *reply* messages, both with same *message format*

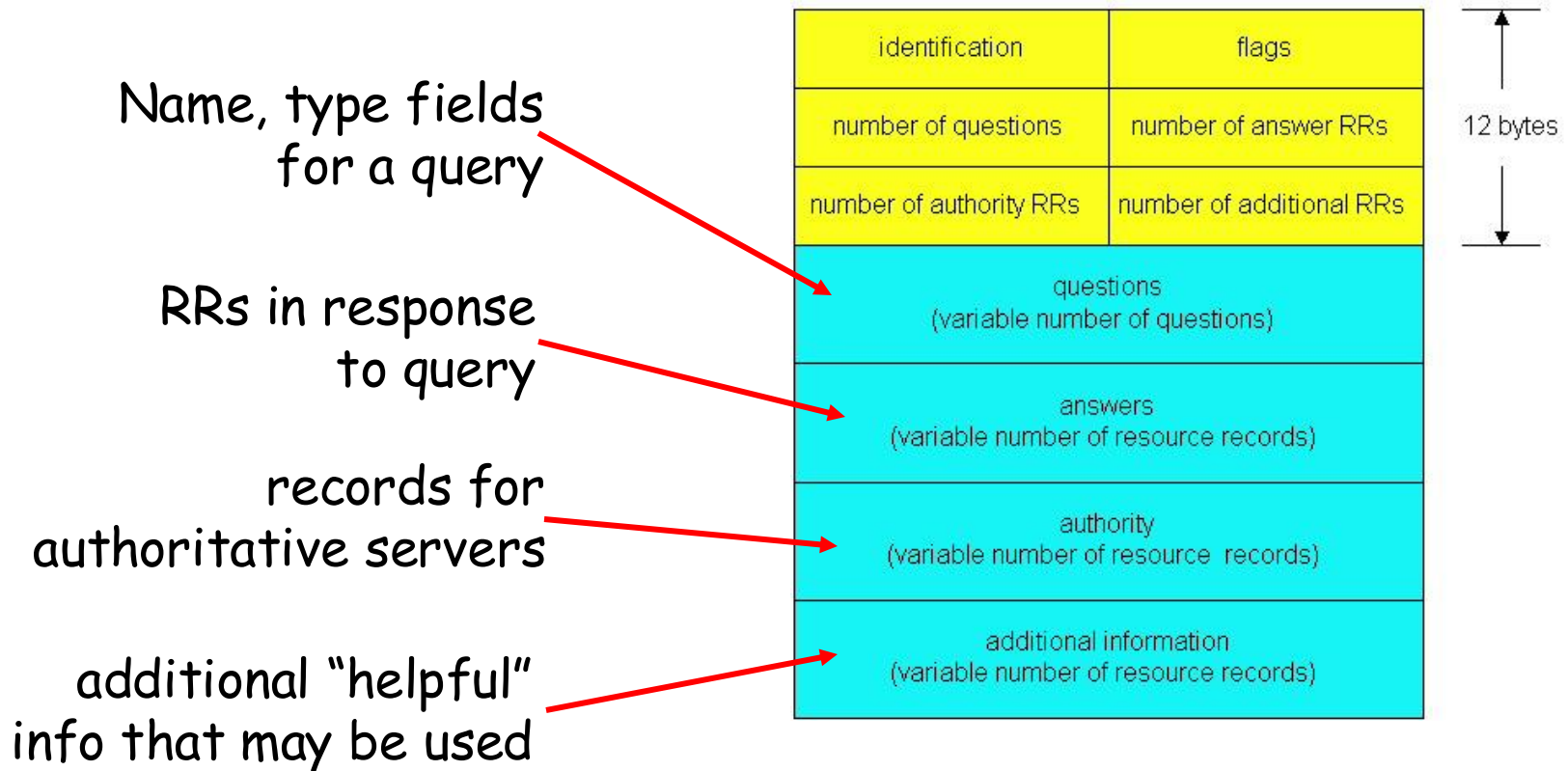
msg header

- ❑ **identification**: 16 bit # for query, reply to query uses same #
- ❑ **flags**:
 - ❖ query or reply
 - ❖ recursion desired
 - ❖ recursion available
 - ❖ reply is authoritative

identification	flags
number of questions	number of answer RRs
number of authority RRs	number of additional RRs
questions (variable number of questions)	
answers (variable number of resource records)	
authority (variable number of resource records)	
additional information (variable number of resource records)	

↑
12 bytes
↓

DNS protocol, messages



Questions ?

Chapter 2: Summary

Our study of network apps now complete!

- Application architectures
 - ❖ client-server
 - ❖ P2P
 - ❖ hybrid
- application service requirements:
 - ❖ reliability, bandwidth, delay
- Internet transport service model
 - ❖ connection-oriented, reliable: TCP
 - ❖ unreliable, datagrams: UDP
- specific protocols:
 - ❖ HTTP
 - ❖ FTP
 - ❖ SMTP, POP, IMAP
 - ❖ DNS
- socket programming

Chapter 2: Summary

Most importantly: learned about *protocols*

- typical request/reply message exchange:
 - ❖ client requests info or service
 - ❖ server responds with data, status code
- message formats:
 - ❖ headers: fields giving info about data
 - ❖ data: info being communicated
- control vs. data msgs
 - ❖ in-band, out-of-band
- centralized vs. decentralized
- stateless vs. stateful
- reliable vs. unreliable msg transfer
- “complexity at network edge”

Questions?